Procedia Computer Science

Volume 53, 2015, Pages 19–28

2015 INNS Conference on Big Data

Managing Vagueness with Fuzzy in Hierarchical Big Data*

Daniel J. Lewis and Trevor P. Martin

University of Bristol, Bristol, England, U.K.
daniel.lewis@bristol.ac.uk, trevor.martin@bristol.ac.uk

Abstract
Out of the web of linked open data, comes a sense of networked "Big Data." This large scale interconnected data is hierarchical, and often messy and full of subjective bias particularly when mass collaboration is concerned (e.g. wikipedia). In this paper we apply fuzzy set theory, specifically the X-μ approach which is shown to be more efficient than a standard fuzzy approach, to attributes within linked data. We look at hierarchical structures, using an example from the films subset of the DBpedia data repository. The hierarchical nature of film categories lends itself well to our application, and we apply fuzzy models to handle the vagueness in attributes such as film length, film budget, and box office takings.

Keywords: Fuzzy Set Theory, X-mu Fuzzy Seta, Big Data, Semantic Web, Linked Data, Open Data

1 Introduction: Why we need fuzzy

Berners-Lee discussed the vision of what became the Semantic Web and Linked Data in 1999 [1]. In this talk he mentioned that humans not only often communicate in a "fuzzy way," but also solve seemingly unrelated problems in an intuitive fashion. In this paper we outline our approach to providing a fuzzy layer on top of Linked Open Data (a form of "Big Data" available openly, often via the world-wide-web).

Data mining over distributed data has been a challenge [2, 3], and the web of linked open data is most likely the largest network of distributed data objects[4]. Due to the "big data" nature of the web of linked open data, we can get insight into interesting relationships between attributes of the objects. The hierarchical, or object-orientated, nature of linked open data, can add value to the insight we can glean. However, such large amounts of interconnected data, made by communities of different humans and/or generated by machines, may create instances where there are mismatches in data structure, conflicts in values and even subjective bias. These inaccuracies may be intentional and purposeful, or they may be unintentional and accidental. As noted by Boyd and Crawford[5] even within the context of big data, subjectivity is common. Fuzzy has proven itself to be useful for handling these vaguenesses and subjective uncertainties, whether that is in systems control, data mining, linguistic summaries or "computing with words".

*supported by EPSRC and BT under CASE award 11220336

Selection and peer-review under responsibility of the Scientific Programme Committee of INNS-BigData2015

doi:10.1016/j.procs.2015.07.275

Figure 1: Map showing the location of London and Gatwick airport, in comparison to the rest of the United Kingdom.

An example of using fuzzy to handle vagueness can be found in systems which handle geographic/spatial data. In Figure 1, for example, "London" and "London Gatwick" are shown. Although there are roughly 28 miles between them, a fuzzy approximation might regard them as "close enough" to be treated as the same location. For example, a prediction of weather at "London Gatwick" will be almost identical to a prediction for in "London". Examples of practical fuzzy usage in temporal and/or geographical systems can be seen in McBratney and Odeh[6], Hansen and Riordan[7], Chen and Hwang[8]. Based on previous experience and the examples provided above, we find that fuzziness can handle some of these inaccuracies. By embracing the vagueness and uncertainties which permeate the linked data web through the use of fuzzy modelling we aim to by-pass the irregularities of data types, and handle the generalisation of certain forms of distributed information.

We have already seen in the introduction how big data is prone to vagueness and uncertainties. The first part of this paper argues that approaching vagueness on big linked data is a human-orientated perspective, it describes the two ways fuzzy can be applied to traditionally crisp data sets, introduces examples of vague and conflicting data within the movie data on DBpedia, and proposes that fuzzy can be used within big data to simplify analysis and data mining. The second part of the paper is example-centric, and asks a question of the data, describes how fuzzy solves this problem, briefly describes the X-μ approach as it is directly mappable on to data queries, and then finally discusses the efficiency of such an approach.

2 Related Work: Two approaches to applying fuzzy to Semantic Web / Linked Data

There are two approaches to adding or modelling fuzziness on the Semantic Web and in Linked Data. The different approaches appertain to differing scenarios, and each has its own benefits and disadvantages.

In the first approach we can accept fuzziness, and build it into ontologies. In this case one could take the approach that fuzzy values should be added into the semantic web data types, and therefore a fuzzy ontology language or a fuzzy resource description framework ("RDF") would be required. An example of this approach is summarised by Straccia[9], who extends the web ontology language ("OWL"), the data framework RDF, as well as semantic web rule and query languages using a generalisation from crisp into fuzzy. It builds on and relates to existing work defined regarding Fuzzy Description Logics by Straccia[10], by Dubois, Mengin and Prade[11] and by Sánchez and Tettamanzi[12][13]. The benefit of this particular approach is that the fuzzy memberships are defined by the person, organisation or system which defines the data type and/or generates the data. The downside is that the definition is either not fully objective, or subjective from the definers perspective therefore indicating bias. As fuzzy membership functions are application and domain specific, i.e. they are problem-specific, this

may not necessarily fit with the broadness of the vision of linked data.

Alternatively, we acknowledge that data is forced into crisp, but subjective, categories, and we are required to build a fuzzy layer after the release of data. In this case one takes the approach that the data on the semantic web, or in the web of linked data, is already full of imprecision, vagueness or uncertain levels of subjectivity. A fuzzy layer can then be applied on top of the existing seemingly crisp web of data through the use of interpretation. Instead of modifying the data types, we add an interpretive layer which is fuzzy. An example of this can be found in the work of Vojtáš[14], who applies fuzziness to a rule engine for search. The benefit of this particular approach is that the fuzzy memberships are defined by the application designer/developer, and are therefore problem-specific. This approach could be seen as similar to that taken in fuzzy control systems[15], where inputs and outputs of the system are crisp, but the internal reasoning engine is fuzzy, through techniques such as Mamdani [16] or Sugeno [17] inference. The downside to this approach is that the membership function might not be defined by an expert in the particular data and data types, and consultancy may need to be sought in production-ready systems. It is worth noting that some work has been done relating Concept Lattices (found in Formal Concept Analysis) with Semantic Web Ontologies[18, 19] and Description Logic. It is clear that the extent of a concept lattice can be related to a class instance, and the intent is the attribute definition. With this in mind, there has also been much work on Fuzzy Concept Lattices, for example[20, 21, 22], and the handling thereof[23]. Although much of this work regards morphing traditional concept lattices into generalised concept lattices, rather than the application-centric approach of applying an additional layer on top of data sets, we can glean important insight into the fuzzification and fuzzy handling of these systems.

3 Preliminaries: Introducing the example

In standard set theory, it is understood that the cardinality of a subset B of a set A, is less than or equal to the cardinality of the set A. The converse could be defined as: $if\ B\ \subset A\ then\ |A|\ >\ |B|$

This is particularly useful when considering categories, and is of interest because the idea of class and category hierarchy appears in the Semantic Web, through the use of classes in ontologies or through the "broader" propositions of the Simple Knowledge Organisation System (SKOS)[24][1].

The "broader" relation defined in SKOS is used within DBpedia[25][2]. The DBpedia system is a service on the web which has crawled the wikipedia website and others, it transformes the data into a structured form using semantic web technologies, and provides it as linked open data with HTML web page views for human agents, and RDF views for machine agents. The SKOS ontology is used in DBpedia to define categories as instances of $SKOS : Concept$, and categories in DBpedia relate to their child categories using the $SKOS : broader$ relation, allowing a hierarchy of categories. The DBpedia system has its own class hierarchy defined in the Web Ontology Language (OWL). An example can be seen in the definition of films on DBpedia. There is a hierarchy of film categories, and Figure 2 shows part of the graph describing genres. The system allows more than one parent category, and could therefore be seen as a multiple inheritance system. From above, the number of films with membership in

[1] SKOS is an ontology which allows us to define "Concepts", which may be "broader", "narrower" than other "Concepts". The SKOS Ontology is a W3C standard, see www.w3.org/2004/02/skos/

[2] More information about the DBpedia system can be found via the DBpedia website: http://dbpedia.org/

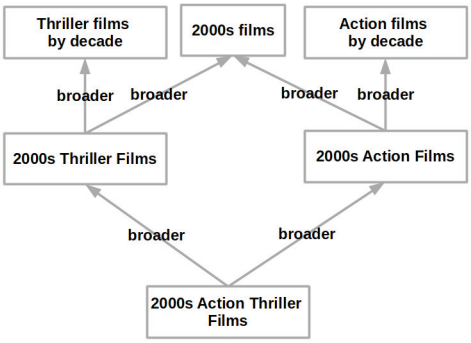


Figure 2: A subset of the film types found in DBpedia using the SKOS ontology. Exemplifying the broader relationship.

one $SKOS : Concept\ A$ is greater than or equal to the number of films with membership of another $SKOS : Concept\ B$ providing that A is $SKOS : broader$ than B.

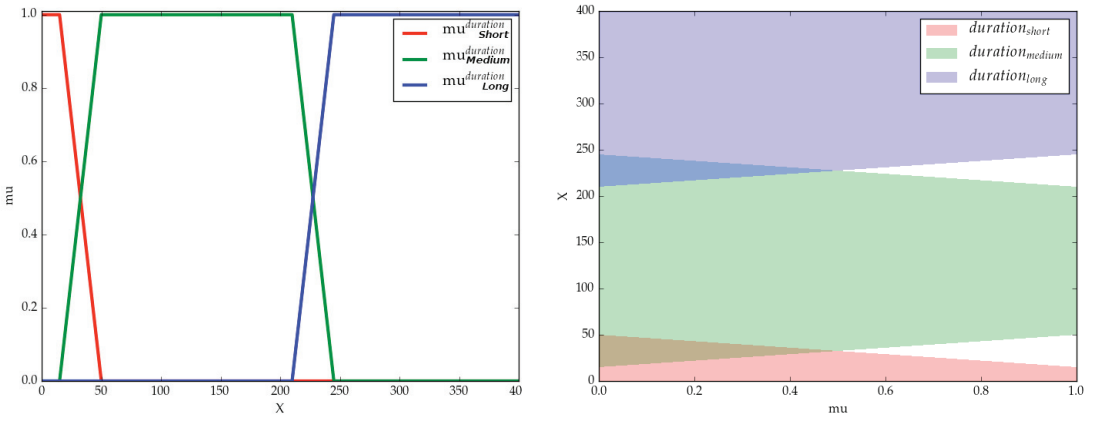
However, there is vagueness, uncertainty, subjectivity and even missing values within the DBpedia film data set, due mainly to the nature of collaborative editing within the wikipedia source. Fuzzy sets can model vagueness and uncertainties[26] in a way that is understandable to a human[27]. Fuzzy attributes in the films example include duration, budget, takings (gross) and even viewer ratings (from an external source). Fuzzy set theory is an extension, or generalisation, to classical set theory where each element within a set (finite or infinite) has a membership value μ within the real number interval $[0, 1]$. Fuzzy memberships are either stored in a lookup table, or have some algebraic membership function for use in symbolic/analytic operations. A fuzzy membership function is defined as: $\mu : U \rightarrow [0, 1]$ where μ is the membership function of a fuzzy set, and U is the domain universe. The membership function of a fuzzy set A is often written $A(x)$ where $x \in U$. The standard definition of fuzzy set theory uses the following operations:

- Fuzzy Negation or Set Complement: $\neg A(x) = 1 - A(x)$ for each item x in set A
- Fuzzy Union: $(A \cup B)(x) = \max(A(x), B(x))$ for each item in both A and B
- Fuzzy Intersection: $(A \cap B)(x) = \min(A(x), B(x))$ for each common item in A and B

The X-μ approach (see below) is an alternative to the traditional fuzzy set theory, and offers useful methods for efficient modelling and analysis of fuzzy data. The remainder of this paper describes the approach using the DBpedia and SKOS ontology as examples, and the results of cardinality and attribute association for each level in the category hierarchy.

4 Vague and conflicting data in the example

An early step in any knowledge engineering or data mining project is that of data cleaning, to handle any inaccuracies, errors and formatting issues. This step is usually partnered with conversion of the data into a usable format, for example the transfer of data in comma-separated-value format into a relational database. This step is often done automatically, and there is potential that not all of these issues are handled. The philosophy of “fuzzy” lends itself well



(a) Traditional fuzzy representation of film duration (b) The X-μ representation of film duration

Figure 3: Traditional and X-μ representations of fuzzy film duration categories: short, medium and long

to some of these issues, as a precise value is of less interest than the more general category to which it belongs. In the DBpedia film example, there are instances of messy and missing data, due to the fact that DBpedia retrieves much of its data by mining the massively collaborative encyclopedia Wikipedia. We can see, for example, conflicting values for attributes, such as:

Film URI	Runtime 1	Runtime 2
<http://dbpedia.org/resource/A_Knight's_Tale>	7920.0	8280.0

In this case, we might take the average of all conflicting runtimes and apply the fuzzy model - so that the data is stored as a fuzzy category such as a medium duration with a certain degree of membership. Formatting issues can arise in film takings, for example:

URI	Gross
<http://dbpedia.org/resource/BloodRayne_(film)>	"3650275.0" ^^< $usDollar$ >
<http://dbpedia.org/resource/Iron_Man_(2008_film)>	"5.85174222$E8$" ^^< $usDollar$ >
<http://dbpedia.org/resource/V2:_Dead_Angel>	"1541266.0" ^^< $euro$ >

Questions of formatting, or precision (for example some films are specific about their takings, some others simply say 1 million, for example), and questions of translating currencies can be resolved by standard approaches. The use of data stored as fuzzy categories of low, medium and high can also be used.

5 A note on the X-μ Approach

The X-μ approach [28, 29, 30] uses the inverse of a membership function. $\mu^{-1} : [0, 1] \rightarrow P(U)$ where $P(U)$ is the powerset of the universe. Whenever possible the symbolic representation of the function is used. An X-μ function on a continuous domain in symbolic form, for instance, is a function returning an interval. In general, the output is a set, which represents the value at any point along the μ domain; this set can be processed by any standard set-theoretic operator. The effect is a result which adheres to the classical set theory laws at any specified membership [30]. As an example, we could use a collection of fuzzy membership functions representing

"short", "medium" and "long" films. According to the Academy of Motion Picture Arts and Sciences [31] a feature length film is any duration from 40 minutes, and we can use this to derive fuzzy definitions for film duration. The standard fuzzy membership functions and the X-μ forms for duration can be seen in Figures 3a and 3b. The comparison between traditional fuzzy methods and the X-μ method shows the retention of the semantics of the original sets. It is also closer to classical set theory, as for whatever value of μ the result is a classical set, and treated with traditional classical set operations. This has efficiency benefits, as classical set operations are less complex than traditional fuzzy set operations, plus when combined with symbolic formulations is simply a case of symbolic manipulation rather than requiring a sampling (or granular) approach. The primary benefit of using the X-μ approach is that it is directly mappable on to data query languages, such as SPARQL, the query language for the Semantic Web and Linked Data.

6 Querying with fuzziness

SPARQL[32], is the query language for Semantic Web and Linked Data, and is similar to SQL (found in relational database management systems). As the X-μ approachreturns a set of values based on some μ value, this can be directly mapped on to a SPARQL query (or any other data query language). For example, if the result of an X-μ function is an interval, the lower and upper bounds can be used as a lower and upper bounds of a query.

6.1 Discovering cardinality on crisp categories with SPARQL Queries

It is understood within both knowledge engineering and software engineering that the number of instances in a class will be greater than or equal to the number of instances within one of its subclasses. This can be seen in the following results generated via a SPARQL query on the DBpedia data set of films. The following listing is a SPARQL query that counts the number of unique films in DBpedia which are an instance of all subcategories of "2000s Films" which we shall label $F1$.

```
SELECT count(DISTINCT ?film) WHERE {
  ?subcategory skos:broader <http://dbpedia.org/resource/Category:2000s_films> .
  ?film dcterms:subject ?subcategory }
```

The result, which we will label $R1$, of the above as executed at the beginning of the year 2015 is 9638 values. To compare we can run the following listing, which is a SPARQL query that counts the number of unique films in DBpedia which are an instance of "2000s Action Films" which we shall label $F2$, a subcategory of "2000s Films" $F1$:

```
SELECT count(DISTINCT ?film) WHERE {
  ?film dcterms:subject <http://dbpedia.org/resource/Category:2000s_action_films> }
```

The result, which we will label $R2$, of the above as executed at the beginning of the year 2015 is 575 values. It is clear that: $R1 = |F1|$ and $R2 = |F2|$ and $F1 \sqsubseteq F2$ therefore $R2 < R1$

6.2 Discovering cardinality on fuzzy categories with SPARQL Queries

We take the short duration X-μ function: $X_{short}^{duration} = [0.0, -35.0\alpha + 50.0]$, and build a SPARQL query based on the interval:

```
SELECT COUNT(DISTINCT ?f_uri) WHERE {
        ?f_uri  dcterms:subject <http://dbpedia.org/resource/Category:2000s_action_films> ;
                        dbpedia-owl:runtime ?runtime .
                        FILTER (?runtime >= 0.0) .
                        FILTER (?runtime <= {(-35.0 α + 50.0)*60}) }
```

Here, we find the count of all distinct URIs of action films released between 2000 and 2009 (inclusive) as indicated by the 2000s_action_film category, we retrieve the runtime, filtering between our lower bound and upper bound of our X-μ interval. Note that the upper bound is multiplied by 60 as our interval is in minutes, whereas the data is in seconds. More importantly, our interval has the symbol α. These symbols will need to be discretised. This is unfortunate as it increases execution time in correlation with the granularisation. In the next section we discuss scalability from an efficiency perspective, in particular a report is given on SPARQL queries from a crisp and from an X-μ perspective.

7 Big Data Scalability

Drilling-down, from a data science perspective, involves a query that includes one or many set operations. We argue that the X-μ approach is more processor efficient, and is therefore more useful for applications that require scalability. In this section we firstly compare practical efficiency between the set operations of traditional fuzzy set theory and the set operations of X-μ. We then deliver an efficiency report on X-μ over SPARQL queries.

7.1 Comparing efficiency of the X-μ approach and the traditional approach to fuzzy

The following is a comparison of the efficiency of X-μ set operations and traditional fuzzy set operations. Calculation times for a test of 50,000 runs per set operation are shown in seconds (rounded). Tests were built using Python[3]. Note that there may be some insignificant outliers in shortest and longest times, due to irrelevant and unavoidable background operations, these are reported nonetheless for general interest. The average values are the significant values for our research. The comparison takes the form of two membership functions M representing medium values over a continuous universe having the interval [1, 6], and L representing large values over the same continuous universe. The results for the X-μ operations for union ($\cup$), intersection ($\cap$) and set difference ($\setminus$) are :

	X-μ $L \cup M$	X-μ $L \cap M$	X-μ $L \setminus M$	X-μ $M \setminus L$
Shortest	0.000649	0.000649	0.002535	0.002335
Average	0.000667	0.000664	0.002586	0.002382
Longest	0.019448	0.018663	0.021275	0.020763

Traditional Fuzzy Operations on the other hand, over the same number of runs (50,000). Note that with traditional fuzzy, we must "chunk" the function into granules, the granularity chosen for these operations is 100 over U. We get the following results in seconds:

	Fuzzy $M \cup L$	Fuzzy $M \cap L$	Fuzzy $L \setminus M$	Fuzzy $M \setminus L$
Shortest	0.010632	0.010668	0.011136	0.011295
Average	0.010955	0.011055	0.011416	0.011613
Longest	0.276682	0.025687	0.026220	0.026597

For the traditional fuzzy operations, if we increase the granularity by a power of 10, i.e. a granularity of 1000, the calculation times increase by a factor of 10. Calculation times in seconds per operation are:

[3]Python 2.7.8 :: Anaconda 2.1.0 (64-bit), on an Intel i7 quad-core 2.10GHz processor, 8GB RAM, Ubuntu Linux version 14.04

	Fuzzy $M \cup L$	Fuzzy $M \cap L$	Fuzzy $L \backslash M$	Fuzzy $M \backslash L$
Shortest	0.107317	0.108812	0.114085	0.116270
Average	0.112935	0.112298	0.117960	0.122728
Longest	1.665969	0.136700	0.128512	0.147879

These granularisations need not occur when performing symbolic X-μ operations, as the operations in X-μ are simply combined using traditional algebraic formulae combination techniques. For this reason set operations in X-μ can be seen as being of O(1) efficiency, whereas traditional fuzzy operations require O(g) efficiency, where g represents the granularisation of the membership function prior to the set operation.

7.2 Efficiency report on X-μ over SPARQL

To set a baseline figure, we must perform a SPARQL query from a crisp perspective. Here we have firm boundaries, in our example we can compare three max runtimes for films separated by 5 minutes (for this example 8100, 8400 and 8700 second durations). We can perform the following crisp SPARQL query where R is our value:

```
SELECT COUNT(DISTINCT ?f_uri) WHERE {
    ?f_uri dcterms:subject <http://dbpedia.org/resource/Category:2000s_action_films> ;
        dbpedia-owl:runtime ?runtime .
        FILTER (?runtime >= 0.0) .
        FILTER (?runtime <= \{$R$\}) }
```

The test is performed on the official DBpedia SPARQL endpoint[4], over a high-bandwidth internet connection using a client-side script developed in the Python programming language. Execution time results are in seconds and are rounded, they are inclusive of HTTP fetch-time and the average is found over 100 iterations.

	[0,8100]	[0,8400]	[0,8700]
Shortest	0.19	0.19	0.19
Average	0.21	0.20	0.20
Longest	0.54	0.49	0.39

We are able to build test a set of SPARQL queries to fetch results using X-μ member functions. Considering the runtime example, we are able to push values of α into the X-μ function to fetch an interval, which we then use the lower and upper bounds as part of our SPARQL query thus.

```
SELECT COUNT(DISTINCT ?f_uri) WHERE {
    ?f_uri dcterms:subject <http://dbpedia.org/resource/Category:2000s_action_films> ;
        dbpedia-owl:runtime ?runtime .
    FILTER (?runtime >= {runtime(α).inf * 60} ) .
    FILTER (?runtime <= {runtime(α).sup * 60} ) }
```

runtime() above could be short, medium or long or a set theoretic combination of those categories such as (*short**medium*). In the following execution tests we execute the SPARQL queries using the short runtime membership function as defined earlier in this paper, and the official DBpedia SPARQL endpoint as above. They are performed with granularities of 10, 100 and 1000. Execution times are for the total of all granules.

	g = 10	g = 100	g = 1000
Shortest	1.81	18.93	176.52
Average	1.98	20.69	179.20
Longest	3.09	24.96	182.04

[4] http://dbpedia.org/sparql

8 Summary

In summary, we have explained the vagueness inherent in hierarchical big data, namely in the linked open data cloud. We have explained how a fuzzy set approach can embrace the vagueness either by applying fuzzy at the ontology level, or by implementing a fuzzy model on top of pre-existing crisp data sets. The X-μ approach is introduced within context of vagueness on linked open data, and is described in terms of its efficiency and semantics. An example X-μ function is then mapped on to a SPARQL query, ready to query the linked open data cloud. Finally, we discussed the scalability of such operations in terms of processing capability. Further work is required in the form of X-μ querying of knowledge-bases, however, the efficiency of X-μ in comparison to traditional fuzzy, and X-μ being mappable to query languages indicates much potential for positive future research and development.

References

[1] T. Berners-Lee, Transcript of a talk given by Tim Berners-Lee to the LCS 35th Anniversary Celebrations, Web page (Apr. 1999).
URL `http://www.w3.org/1999/04/13-tbl.html`

[2] Y. Fu, Distributed data mining: An overview, Newsletter of the IEEE Technical Committee on Distributed Processing (2001) 5–9.

[3] B.-H. Park, H. Kargupta, Distributed Data Mining: Algorithms, Systems, and Applications, in: Handbook of Data Mining (Human Factors and Ergonomics Series), Taylor and Francis, 2002, pp. 341–358.

[4] C. Bizer, T. Heath, T. Berners-Lee, Linked Data - The Story So Far, International Journal on Semantic Web and Information Systems 5 (3) (2009) 1–22. `doi:10.4018/jswis.2009081901`.

[5] d. boyd, K. Crawford, Six Provocations for Big Data, SSRN Scholarly Paper ID 1926431, Social Science Research Network, Rochester, NY (Sep. 2011).

[6] A. B. McBratney, I. O. Odeh, Application of fuzzy sets in soil science: fuzzy logic, fuzzy measurements and fuzzy decisions, Geoderma 77 (2) (1997) 85–113.

[7] B. K. Hansen, D. Riordan, Weather prediction using case-based reasoning and fuzzy set theory, Ph.D. thesis, Citeseer (2000).

[8] S.-M. Chen, J.-R. Hwang, Temperature prediction using fuzzy time series, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 30 (2) (2000) 263–275.

[9] U. Straccia, Foundations of Fuzzy Logic and Semantic Web Languages, CRC Press, 2013.

[10] U. Straccia, A fuzzy description logic for the semantic web, Capturing Intelligence 1 (2006) 73–90.

[11] D. Dubois, J. Mengin, H. Prade, Possibilistic uncertainty and fuzzy features in description logic. A preliminary discussion, Capturing Intelligence 1 (2006) 101–113.

[12] D. Sánchez, A. G. Tettamanzi, Generalizing quantification in fuzzy description logics, in: Computational Intelligence, Theory and Applications, Springer, 2005, pp. 397–411.

[13] D. Sánchez, A. G. Tettamanzi, Fuzzy quantification in fuzzy description logics, Capturing intelligence 1 (2006) 135–159.

[14] P. Vojtáš, Fuzzy logic aggregation for semantic web search for the best (top-k) answer, Capturing Intelligence 1 (2006) 341–359.

[15] K. M. Passino, Fuzzy Control, Addison-Wesley, 1998.

[16] E. H. Mamdani, S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, International journal of man-machine studies 7 (1) (1975) 1–13.

[17] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, Systems, Man and Cybernetics, IEEE Transactions on SMC-15 (1) (1985) 116–132.

[18] G. Stumme, A. Maedche, Ontology merging for federated ontologies on the semantic web, in: Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001), Citeseer, 2001, pp. 413–418.

[19] A. Maedche, Ontology learning for the semantic web, Springer Science & Business Media, 2002.

[20] R. Belohlávek, V. Vychodil, What is a fuzzy concept lattice, in: Proc. CLA, Vol. 5, CEUR, 2005, pp. 34–45.

[21] Q. T. Tho, S. C. Hui, A. C. M. Fong, T. H. Cao, Automatic fuzzy ontology generation for semantic web, Knowledge and Data Engineering, IEEE Transactions on 18 (6) (2006) 842–856.

[22] A. Majidian, T. P. Martin, M. E. Cintra, Fuzzy formal concept analysis and algorithm, in: Proceedings of the 11th UK Workshop on Computational Intelligence, Citeseer, 2011, pp. 61–67.

[23] T. P. Martin, N. H. A. Rahim, A. Majidian, A general approach to the measurement of change in fuzzy concept lattices, Soft Computing 17 (12) (2013) 2223–2234. doi:10.1007/s00500-013-1095-6.

[24] A. Miles, S. Bechhofer, SKOS Simple Knowledge Organization System Reference, Web page (Aug. 2009).
URL http://www.w3.org/TR/skos-reference/

[25] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, others, DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia, Semantic Web.

[26] D. Dubois, H. Prade, The three semantics of fuzzy sets, Fuzzy sets and systems 90 (2) (1997) 141–150.

[27] L. A. Zadeh, Fuzzy logic= computing with words, Fuzzy Systems, IEEE Transactions on 4 (2) (1996) 103–111.

[28] T. P. Martin, B. Azvine, The X-mu approach: Fuzzy quantities, fuzzy arithmetic and fuzzy association rules, in: 2013 IEEE Symposium on Foundations of Computational Intelligence (FOCI), IEEE, 2013, pp. 24–29. doi:10.1109/FOCI.2013.6602451.

[29] D. J. Lewis, T. P. Martin, X-mu Fuzzy Association Rule Method, in: Proceedings of the 13th UK Workshop on Computational Intelligence (UKCI) 2013, IEEE, 2013, pp. 144–150. doi:10.1109/UKCI.2013.6651299.

[30] D. J. Lewis, T. P. Martin, The X-mu Approach: In Theory and Practice, in: A. Laurent, O. Strauss, B. Bouchon-Meunier, R. R. Yager (Eds.), Information Processing and Management of Uncertainty in Knowledge-Based Systems, Vol. 444 of Communications in Computer and Information Science, Springer, Heidelberg, 2014, pp. 406–415. doi:10.1007/978-3-319-08852-5_42.

[31] A. Awards, Rules of the 87th Academy Awards of Merit, Web page (2015).
URL http://www.oscars.org/sites/default/files/87aa_rules.pdf

[32] W3C, SPARQL 1.1 Overview, Web page (Mar. 2013).
URL http://www.w3.org/TR/sparql11-overview/